NAVAL POSTGRADUATE SCHOOL

MONTEREY, CALIFORNIA

THESIS

COORDINATION AND CONTROL FOR MULTI-QUADROTOR UAV MISSIONS

by

Levi Jones

March 2012

Thesis Advisor: Oleg Yakimenko
Second Reader: Noël Du Toit

Approved for public release; distribution is unlimited

THIS PAGE INTENTIONALLY LEFT BLANK

REPORT DOCUMENTATION PAGE		Form Approved OMB No. 0704–0188

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	March 2012	Master's Thesis

4. TITLE AND SUBTITLE Coordination and Control for Multi-Quadrotor UAV Missions	5. FUNDING NUMBERS
6. AUTHOR(S) Levi Jones	
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Naval Postgraduate School Monterey, CA 93943–5000	8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING /MONITORING AGENCY NAME(S) AND ADDRESS(ES) N/A	10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government. IRB Protocol number _____N/A_____.

12a. DISTRIBUTION / AVAILABILITY STATEMENT	12b. DISTRIBUTION CODE
Approved for public release; distribution is unlimited	

13. ABSTRACT (maximum 200 words)

This thesis is centered upon an optimal trajectory generation algorithm that allows real-time control for cooperation of multiple quadrotor vehicles for intelligence, surveillance, and reconnaissance missions with minimal user input. The algorithm is designed for an indoor environment where global positioning system data is unavailable or unreliable, forcing the vehicles to obtain position data using other sensors. This thesis specifies the lab setup and well as the control approach used. Data acquired from two experiments is included to demonstrate the effectiveness of the control approach.

The control approach described within allows for a fully autonomous system with user input required only at the initiation of a mission. The algorithm blends trajectory planning, trajectory following, and multi-vehicle coordination to achieve the goal of autonomy. The focus of the thesis was on trajectory generation and multi-vehicle coordination, while leveraging existing trajectory following controller implementations. The trajectory generation is accomplished with a direct transcription of the optimization problem that leverages inverse dynamics and separates spatial and temporal planning. The vehicle motion is constrained, and simplifying multi-vehicle coordination assumptions allow for the efficient solution and execution of the problem.

14. SUBJECT TERMS Quadrotor, Cooperation, Quanser, LQR, Direct Methods, Trajectory Generation, Trajectory Following, Optimization, Inverse Dynamics, IDVD.	15. NUMBER OF PAGES 89
	16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	UU

THIS PAGE INTENTIONALLY LEFT BLANK

Approved for Public Release; distribution is unlimited

COORDINATION AND CONTROL FOR MULTI-QUADROTOR UAV MISSIONS

Levi Jones
Lieutenant, United States Navy
B.S., Miami University, 2007

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN MECHANICAL ENGINEERING

from the

NAVAL POSTGRADUATE SCHOOL
March 2012

Author: Levi Jones

Approved by: Oleg Yakimenko
Thesis Advisor

Noël Du Toit
Second Reader

Knox Millsaps
Chair, Department of Mechanical and
Aerospace Engineering

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

This thesis is centered upon an optimal trajectory generation algorithm that allows real-time control for cooperation of multiple quadrotor vehicles for intelligence, surveillance, and reconnaissance missions with minimal user input. The algorithm is designed for an indoor environment where global positioning system data is unavailable or unreliable, forcing the vehicles to obtain position data using other sensors. This thesis specifies the lab setup and well as the control approach used. Data acquired from two experiments is included to demonstrate the effectiveness of the control approach.

The control approach described within allows for a fully autonomous system with user input required only at the initiation of a mission. The algorithm blends trajectory planning, trajectory following, and multi-vehicle coordination to achieve the goal of autonomy. The focus of the thesis was on trajectory generation and multi-vehicle coordination, while leveraging existing trajectory following controller implementations. The trajectory generation is accomplished with a direct transcription of the optimization problem that leverages inverse dynamics and separates spatial and temporal planning. The vehicle motion is constrained, and simplifying multi-vehicle coordination assumptions allow for the efficient solution and execution of the problem.

THIS PAGE INTENTIONALLY LEFT BLANK

TABLE OF CONTENTS

LIST OF FIGURES

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF TABLES

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF ACRONYMS AND ABBREVIATIONS

UV	Unmanned Vehicle
UAV	Unmanned Air Vehicle
UGV	Unmanned Ground Vehicle
USV	Unmanned Surface Vehicle
UUV	Unmanned Underwater Vehicle
ISR	Intelligence, Surveillance, and Reconnaissance
VTOL	Vertical Takeoff and Landing
IMU	Inertial Measurement Unit
LQR	Liner Quadratic Regulator
QuarC	Quanser Real-time Control
DAC	Data Acquisition Card
I/O	Input/Output
PID	Proportional Integral Derivative
ESC	Electronic Speed Controller
GPS	Global Positioning System
LED	Light Emitting Diode
TCPIP	Transmission Control Protocol/Internet Protocol
HIL	Hardware in the Loop
PWM	Pulse Width Modulation
USB	Universal Serial Bus
LTP	Local Tangent Plane
DOF	Degrees of Freedom

THIS PAGE INTENTIONALLY LEFT BLANK

ACKNOWLEDGMENTS

I would like to acknowledge Dr. Oleg Yakimenko and Dr. Noel DuToit for their continuous help and motivation during the thesis process. I would also like to thank Cameron Fulford from Quanser Academic for giving technical assistance on the Quanser quadrotors.

THIS PAGE INTENTIONALLY LEFT BLANK

I. INTRODUCTION AND OVERVIEW OF EXISTING APPROACHES

A. GENERAL

An unmanned vehicle (UV) is a power-driven vehicle that does not carry an operator and is either remotely or autonomously controlled. Unmanned vehicles can perform a wide variety of missions and carry a vast array of payloads. Most are designed to be recovered after use while others are designed to be cheap throwaway alternatives.

UVs are typically broken down into four categories: unmanned air vehicles (UAV), unmanned ground vehicles (UGV), unmanned surface vehicles (USV), and unmanned underwater vehicles (UUV). There has been a recent explosion in the number and variety of designs for each of the preceding types of vehicles, and for good reason. UVs are perfect for performing tasks historically done by humans that are dull, dirty, and dangerous [1].

Examples where UVs are replacing humans are intelligence surveillance and reconnaissance (ISR) missions, collecting signals intelligence, and mapping ocean floor topographies. Each of the previous missions fall in the categories of dull, dirty, and dangerous, and replacing a human with an unmanned vehicle improves safety and efficiency in each case. Instead of confining a pilot to an enclosed cockpit of a spy-plane for hours on end, where fatigue would become problematic for the pilot, an unmanned air vehicle is introduced and the potential for fatigue is reduced. In this case, operators can easily be rotated in shifts without the need for the platform to

return to an airfield. If we take the evolution even further, an autonomous vehicle can take off, perform the mission, and then return to its airfield with little to no human interaction required.

B. VEHICLE SELECTION

When selecting a type of UV for a specific mission, the choices can be daunting, and the options are only expanding with the recent amplification of research into the field of unmanned systems. To name a few options, there are tracked, wheeled, and legged ground vehicles [2]-[4]. These vehicles are accompanied by an array of propeller driven and jet-drive surface craft or underwater vehicles [5]-[6]. In addition, one could choose a blimp, fixed wing or rotary aircraft to conduct a mission [6]-[7]. Although each type of vehicle has its advantages, the quadrotor is the most versatile, substantiated below.

1. Quadrotor Advantages

a. Hover Capability

Quadrotors have the ability to hover in one location for an extended period of time. This hover has several consequences. First, a hover gives the user a great deal of maneuvering flexibility when conducting ISR missions. The hover capability also allows the vehicle to maneuver in a physically constrained environment such as that found in urban areas. The ability to hover enables the vehicle to take off and land vertically, freeing the user from operational constraints typically experienced with fixed wing aircraft due to the reliance on airfields or

large clearings. The vertical takeoff and landing (VTOL) capability also eliminates any need for and launching or recovery equipment.

b. Speed

Although the quadrotor is not as fast as a fixed-wing aircraft, it does have a speed advantage over many other vehicles such as blimps and ground vehicles. The ability to take a direct route to an area of interest also gives it an advantage over ground vehicles that may be forced to avoid obstacles such as plant growth.

c. Size

Current technology levels in batteries and electric drive motors allow us to build quadrotors that are small enough to fit through doorways and maneuver indoors (although collecting sensor data indoors can be problematic) due to the loss of GPS data. This ability to fit through doorways gives the user more flexibility when deploying the vehicle in an urban environment. While employed in a tactical scenario, the size of the quadrotor also means that it has more survivability and a smaller chance of detection by enemy forces.

d. Mechanical Simplicity

Standard helicopters use a tail rotor to balance the torque created by the single rotor head and use a mechanically complex rotor hub to change the pitch of the blades on the main and tail rotors. On the other hand, quadrotors use counter-rotating propellers to eliminate the torque produced by the blades, negating the need for a tail

rotor. Maneuvering a quadrotor is accomplished by changing motor speeds; an approach that is mechanically simple compared to changing the pitch of the blades using a rotor hub. Because the quadrotor uses four sets of blades, each has a smaller diameter to an equivalent sized helicopter. A smaller set of blades possess much less kinetic energy for the same lifting force, reducing the potential for damage to occur in the event of a collision.

2. Quadrotor Disadvantages

Although quadrotors have many advantages, they also have a few drawbacks. The use of continual battery power during flight limits mission durations. The nature of some missions such as signals intelligence may allow the vehicle to land while loitering, battery life still limits range in this case. Quadrotors are also sensitive to disturbances such as wind gusts or rotor wash from nearby aerial vehicles. Payload restrictions also limit the size and number of sensors the vehicle is able to carry.

C. RELATED WORK

1. General

Many universities are using quadrotors in their curricula and conducting research for control and trajectory generation for these vehicles. Individual companies have also started to develop their own quadrotor systems, most aimed at the commercial market.

2. University of Pennsylvania

When you type in “quadrotor” for a Google search, the first website you find is a Wikipedia article, the second is a Youtube video showing the University of Pennsylvania’s (UPenn) quadrotors making very aggressive maneuvers in their indoor lab. The UPenn uses a quadrotor developed by Ascending Technologies to verify their trajectory generation and control algorithms. The UPenn quadrotor system uses an external localization system (VICON) which consists of 20 cameras and the onboard inertial measurement unit (IMU) for state estimation [8]. The video shows that they were able to perform up to three flips, ascend and descend through a narrow window orientated horizontally, and perch on an inverted surface [9]. The trajectories were made possible by using different controllers at different stages of the maneuver.

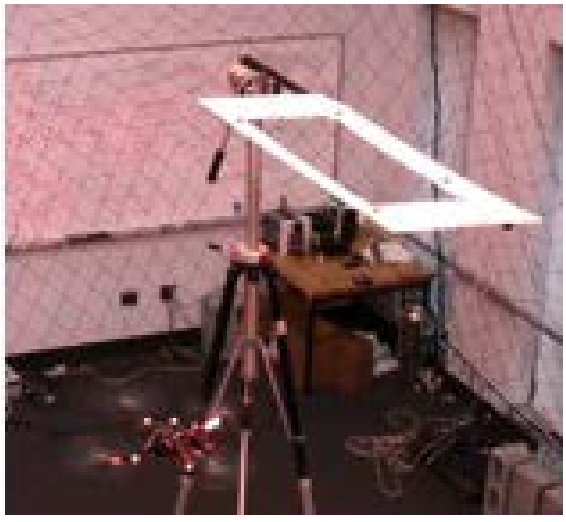

Figure 1. University of Pennsylvania Quadrotor after Descending through an Open Window

Figure 2. University of Pennsylvania Quadrotor Perched on an Inverted Surface

UPenn has also been experimenting with micro UAVs; unmanned vehicles that are between 0.1–0.5 meters in size and 0.1–0.5 kilograms [10]. UPenn uses these micro UAVs for research into formation flying [4] and building structures using the vehicles. UPenn uses a leader-follower approach to formation following where the leader may be a real or virtual vehicle [11]. In order to simulate a construction task, multiple micro UAVs were used to cooperatively transport relatively large blocks of wood along a three-dimensional trajectory [12]. The micro UAVs used in the previous setups also use the Vicon camera setup and an IMU to determine the states of the micro UAVs [11].

3. Standford University

Compared with UPenn, Stanford has taken a different approach to their quadrotor program. Stanford has developed their own vehicle named Stanford Testbed of Autonomous Rotorcraft for Multi-Agent Control (STARMAC). STARMAC uses a 400MHz processor with global positioning system (GPS), an

IMU, and sonar altitude sensors. Stanford's research has focused on outdoor applications, specifically studying the effects on vehicle flight due to vehicular velocity, angle of attack, and airframe design [13].

4. Draganfly Innovations Quadrotor

Draganfly Innovations, Inc. is a based out of Saskatoon, Canada and manufactures different rotorcraft, each with a different rotor setup. Draganflyers are equipped with a full suite of sensors including gyroscopes, accelerometers, and barometric pressure sensors for state estimation. Their standard quadrotor has the capability to carry a 0.25 kg payload and interface with three different cameras [14]. Law enforcement agencies have found a use for the Draganflyer for crime scene investigations. Draganfly Innovations also has customers in the industrial circle as well as photographers and universities.

Figure 3. Draganflyer X-4

Researchers at the Massachusetts Institute of Technology (MIT) use the Draganflyer to investigate swarm techniques that will allow continuous operations using multiple vehicles in a dynamic environment. MIT's mission leads them to focus on health monitoring systems, mission task coordination, and a user interface that supports continuous daily activities [15].

5. Parrot Drone

A prime example of an affordable commercial quadrotor is developed as a toy is built by Parrot, a company better known for electronics such as car stereos. Parrot's AR.Drone uses a Wi-Fi signal that links to an iphone, iPod Touch, iPad, Android smartphone, or a PC using the Linux operating system [16]. The AR.Drone sends a video feed from a forward-looking camera located on the quadrotor that can be displayed on the controlling device. The vehicle uses an IMU and an onboard sonar sensor to determine vehicles states, combined with a feature tracking algorithm that utilizes the downward looking camera for hovering. In addition to normal flight mode, Parrot has developed an application that allows pilots to link with other AR.Drone owners and dogfight. The application uses visual recognition software to determine if a "hit" was scored on the opposing player's brightly colored hull. Although the AR.Drone does have a hovering capability that allows the vehicle to hold altitude and position, it does not incorporate much autonomy in the design and requires constant inputs from a pilot during forward flight.

Figure 4. Parrot's AR.Drone with the Outdoor "Hull"

D. MOTIVATIONS

Unmanned systems are the leading edge of a lot of development in the military and civilian sectors for many reasons. Unmanned systems have the ability to remove human operators from the aircraft, thus removing them from danger. Removing the human operator from the cockpit also removes the need for a single pilot to endure long missions. Instead of one pilot at the controls, a shift of operators can monitor the vehicle and change out personnel to reduce the monotony and fatigue that accompanies long missions. The removal of the pilot also means a reduction in training and re-certification of pilots and operators. If an unmanned vehicle can land itself on a surface ship at night without the need for human interference, there is no need for the waste of fuel and mission resources for a human to practice the procedure.

With the rise in use of unmanned systems and the decrease in defense budgets, the next evolutionary step is

an increased use of autonomy. An autonomous system will decrease the need for operator interaction with the vehicle. This reduction in human interaction with the vehicle means that multiple systems can be operated by one individual for a swarm attack, surveillance from multiple angles, or a sustained surveillance mission using shifts of vehicles. Alternatively, a single operator could operate a smaller number of vehicles and still be free to perform another task in the field.

E. OBJECTIVE

The objective of this thesis is increasing the autonomy of a fleet of homogenous unmanned vehicles, specifically quadrotors used in unison. First, a trajectory generator will be designed for a single vehicle. Once the single-vehicle approach has been verified, the method will be modified for use with multiple vehicles to ensure de-confliction of trajectories to avoid collisions.

II. QUANSER LAB SETUP

A. INTRODUCTION

The laboratory where all trials were completed was designed in an open architecture and reconfigurable fashion. The lab provides an indoor area where all environmental factors can easily be controlled. All Quanser equipment can be controlled in the lab, including the Qball-X4 quadrotor used for demonstration purposes in this thesis.

The ground station is positioned at the edge of the room to allow for a large area in the center where vehicles can be maneuvered. The floor is covered in a thin foam material to eliminate optical reflections from the floor that may cause disturbances with the Optitrack system.

A host model is run on the ground station computer that collects localization data from the Optitrack infrared camera system and joystick using Matlab Simulink. The host model is started first and sends all localization and joystick data to the vehicles via an ad-hoc wireless network.

The control model is built in Matlab Simulink, then downloaded and compiled into an executable onboard the embedded Gumstix computer. The Gumstix computer is connected to the HiQ data acquisition card (DAQ) that encompasses an IMU and an avionics input/output (I/O) card [17].

B. QUANSER REAL-TIME CONTROL SOFTWARE

The Quanser Real-Time Control (QuaRC) software allows for rapid prototyping and testing of control software on the Qball and Qbot. The use of Simulink enables the user to skip the low-level programming and focus on controller design. The QUARC package comes with a Simulink menu option as well as an additional blockset in the Simulink library browser to interface with all Quanser products as well as the Optitrack camera system.

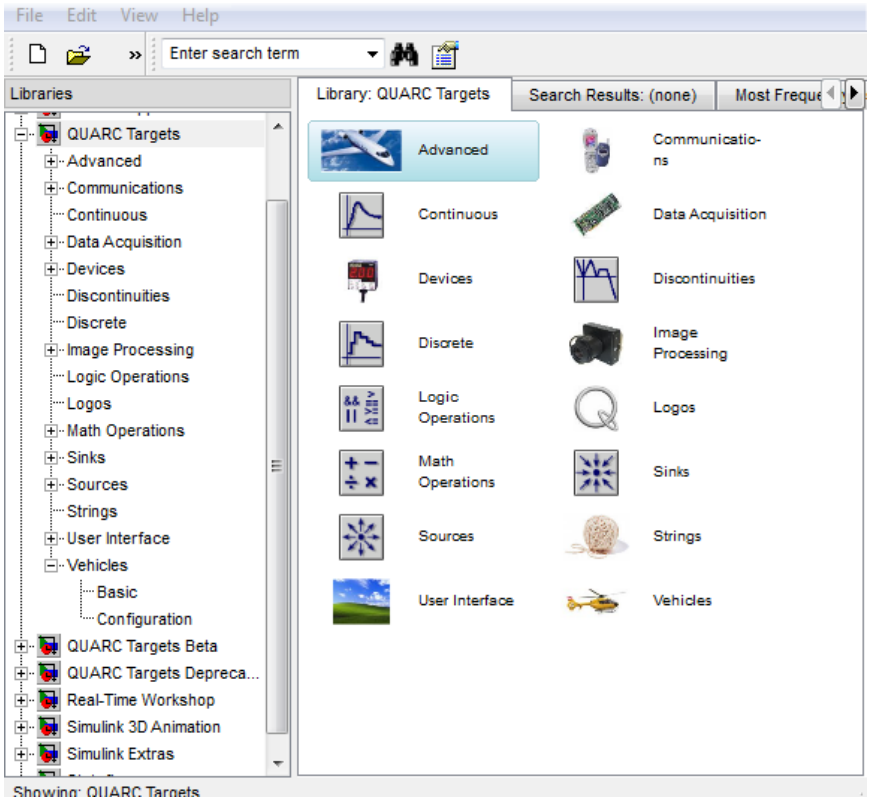

Figure 5. Quanser Simulink Blockset

Simulink files are run under external mode to allow for the code to be built and run on a target vehicle and allows it to use more computing resources so that it can be run at a higher update rate. Multiple models can be run

from the same computer, allowing a single ground station to operate multiple vehicles. The design of the QuaRC software allows the user to modify certain aspects of the code such as changing gains while the code is running. This ability to tune gains on the fly greatly reduces time spent compiling and allows to tune controller gains in flight.

C. QBALL-X4

1. Introduction

The Qball-X4 is a quadrotor helicopter that is approximately 0.7 meters in diameter with an external carbon fiber protective cage. The quadrotor is powered by four motors mounted to the cross-body that each turn a 10-inch fixed-pitch propeller. The vehicle is controlled by an onboard Gumstix computer and is powered by two triple cell 11.1 volt lithium polymer batteries (Figure 10) [17].

2. Main Components

a. Protective Cage and Frame

The external cage is made from thin carbon fiber rods that have been glued into plastic connectors. The cage is assembled into a shape similar to a truncated icosahedron. The shape provides rigidity when the quadrotor is resting on the ground, yet is weak enough that it will absorb the shock from an impact if it were to happen. If the joints are glued properly, the carbon fiber rods simply pull out of their sockets in the plastic connectors during a crash and are easily glued back in place with a hot glue gun. The cage is recessed at the bottom to allow for a stable landing and to houses a sonar sensor for altitude measurements.

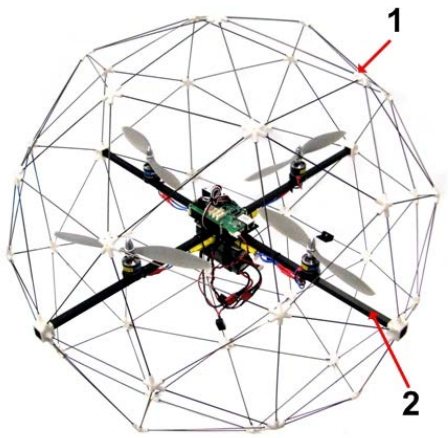

Figure 6. Qball-X4 Cage and Frame

The frame is an aluminum crossbeam structure that connects directly to the protective cage via a small rubber mount in order to reduce shock loading in the event of a collision or hard landing. The frame serves as a rigid mount for all four motors as well as the HiQ DAQ and batteries.

b. Gumstix Embedded Computer and DAQ

The HiQ is the data acquisition card that, along with the Gumstix embedded computer, is responsible for control of the vehicle and reading on-board sensors. The HiQ runs a Linux operating system and has several high resolution sensors [17] for vehicle control and provide the following interfaces:

- 10 pulse width modulated (PWM) outputs for motor control
- 3-axis gyroscope
- 3-axis accelerometer
- 6 analog inputs

- 3-axis magnetometer
- 8 channel RF receiver input
- 4 Maxbotix sonar inputs
- 2 pressure sensors (absolute and relative pressure)
- 11 reconfigurable digital I/O
- 2 TTL serial ports
- Serial GPS input

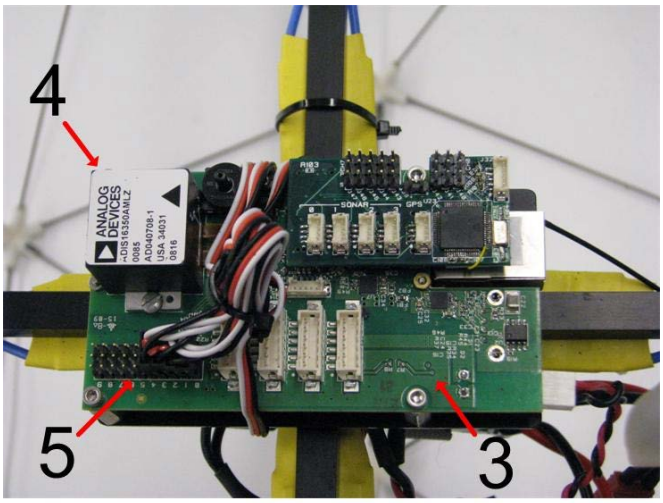

Figure 7. HiQ DAQ

c. Sensors

For many reasons, not all listed sensors were used in the control model. The magnetometer is installed on the HiQ DAW and has an accuracy of 0.5 mGa/LSB. During testing it was determined that the magnetometer was unreliable, presumably because of magnetic fields from nearby electrical wires in the lab. Since the magnetometer was deemed unusable, only the gyroscope and accelerometer

were used for control of roll, pitch, and yaw. The accelerometer has a resolution of 3.33 mg/LSB and the gyroscope is reconfigurable for ±75°/s, ±150°/s, or ±300°/s with a resolution of 0.125°/s/LSB at a setting of ±75°/s [17].

When analyzing the height model, it was determined that the sonar gave a very accurate estimate of the altitude, and was deemed the sensor of choice. The sonar used is a Maxbotix XL-Maxsonar EZ3 that is capable of measuring altitudes between 20 cm and 765 cm with 1cm resolution [17]. The sonar is fixed to the bottom of the protective cage, so a correction must be made for the height difference between the location of the sensor and the center of gravity of the vehicle where the body-fixed coordinated frame is located. Another specification to note here is that the sonar sensor measures a relative height above ground and will give an incorrect reading if another object is below the sensor, such as another vehicle or other obstacle.

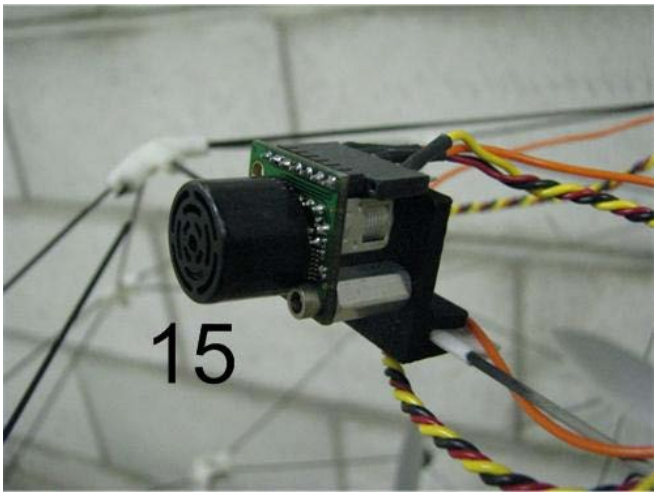

Figure 8. Sonar Sensor

Lastly, since GPS is unavailable or unreliable indoors, no onboard sensors for horizontal position were used. Instead, an external infrared camera tracking system called Optitrack was used and position data was few to the vehicle over an ad-hoc wireless connection from the ground station. The Optitrack system uses infrared cameras and light emitting diodes (LEDs) to track the position of one or multiple reflectors located on the cage of the vehicle. The Optitrack system will be covered in depth in a follow-on section.

Figure 9. Reflector for the Optitrack System

d. Motors and Propellers

The quadrotor is powered by four E-Flite Park 400 motors [18]. The motors each turn a 10x4.7 propeller. The propellers are designed so that the front and back propellers spin in a clockwise direction and the left and right propellers spin counterclockwise. The opposing rotational direction balances the yawing moment created by

the spinning propellers and gives the quadrotor control over yaw angle with and decoupling it from other state variables.

Each motor is controlled by an electronic speed controller (ESC,) shown in Figure 10. Each ESC is connected to the HiQ via the PWM portion of the I/O card on the HiQ. The HiQ sends a command to the ESC between 1ms and 2ms. A command of 1ms corresponds to minimum throttle and a command of 2ms corresponds to maximum throttle.

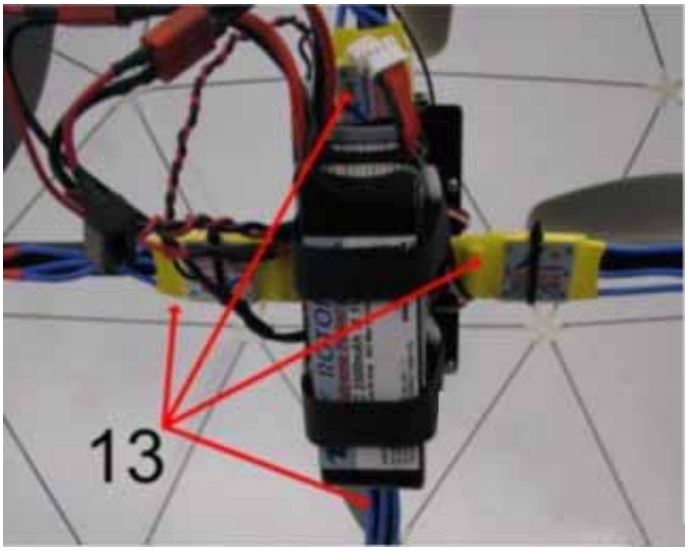

Figure 10. ESCs and Batteries

3. Communication

The Qball-X4 and Qbot are designed to be controlled via a peer-to-peer transmission control protocol/Internet protocol (TCP/IP) wireless Internet connection, but other protocols may be used. Because this thesis involves communication between multiple vehicles, the user datagram protocol (UDP) was selected. A UDP connection allows data to be sent between systems without first establishing a

connection between the two systems [1]. The ability to send and receive data without first establishing a connection gives a lot of flexibility when testing models. In the case of this thesis, using a UDP connection allowed the testing of one vehicle at a time followed by a test of multiple vehicles without altering reconfiguring communication setup in the models.

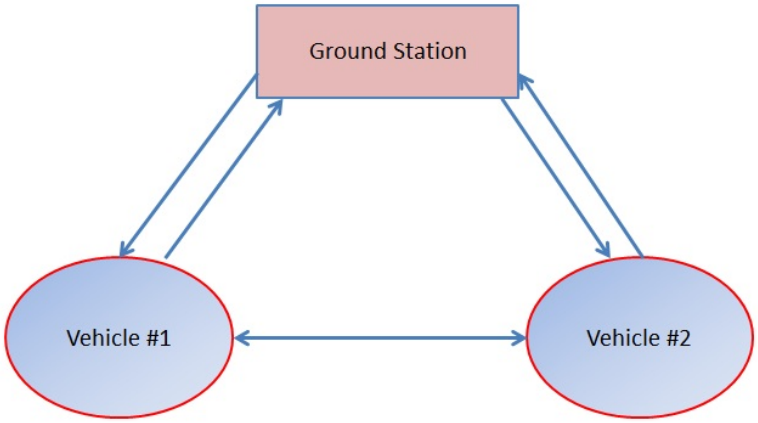

Figure 11. Connection Diagram

The wireless Internet connection on the ground station is established by utilizing a USB Cisco Linksys G type wireless adapter. After a Quanser vehicle is booted up, it automatically creates a network named “GSAH.” Once the vehicle has created the GSAH network, the user connects via standard Windows procedures. Next, the Internet protocol (IP) address is typically pinged to verify a connection to the vehicle before attempting to load a control model. When the control model is run, the vehicle starts sending data if it is setup to do so. When multiple vehicles are used, each is setup to transmit on a different port number and can transmit any data available such as trajectory or current position. The UDP connection allows the quadrotor

to transmit this data, and any other system connected to the GSAH network listening on the correct port will receive the data. This configuration allows a large swarm of vehicles to communicate without unnecessarily complex communication networks.

D. OPTITRACK MOTION CAPTURE SYSTEM

1. Introduction

The Optitrack camera tracking system built by Natural Point, uses infrared LEDs and cameras to track the position of a set of reflectors attached to the quadrotor. The system is run by the ground station computer, where the vehicle coordinates are calculated, then sent to the vehicle. The Optitrack system is necessary because GPS position data is unavailable in the indoor environment.

2. Camera Setup

The Optitrack system has the ability to use up to 24 cameras to capture up to 32 rigid bodies at 100 Hz with an accuracy of 1 cm [20]. The laboratory setup used 11 V100:R2 cameras that each have a field of view of 46 degrees. Each camera has a resolution of 640x480 pixels at a frame rate of 100 frames per second. The cameras were mounted approximately ten feet from the laboratory floor at the edges of the room to give them the largest capture volume possible. The capture volume where the system can track a reflector is approximately 10 feet tall and 12 by 18 feet in width.

Figure 12. V100:R2 Infrared Camera

3. Tracking Tools Software

The Tracking Tools software is used with the Optitrack system and manages the configuration of cameras and distinct reflector patterns used on each vehicle. To calibrate the system, a wand (included with the Optitrack package) is used to triangulate the position of each of the cameras. After opening the Tracking Tools software, the option to create a new calibration file is selected. A tool within the software allows the user to mask any reflections caused by objects in the field of view of the cameras so that they don’t interfere with tracking the reflectors on the vehicles. Next, the “Start Wanding” button is selected and the user moves the wand through the capture volume until the Tracking Tools software indicates that each camera has reached the desired accuracy. Figure 13 shows the view from one camera during the wanding process, note the solid red squares are masks for reflective objects in the capture volume. Once the wanding is complete, the user

selects the “Apply Result” button and the software calculates the relative position and orientation of each camera. Once this has finished, the user places a ground plane tool in the center of the capture volume on a level surface and the software sets this as the ground plane and numeric origin where all measurements will relate to. Once this is complete, the calibration is saved so that it may be referred to later when setting up the host ground station model. If multiple vehicles will be used, a file must be created that allows the software to distinguish the vehicles from one another. To accomplish this, the vehicles are placed in the capture volume, the reflectors are selected, and the “create trackables” button is selected through the right click menu. This trackable file is now saved and referred to later in the host ground station model. After the calibration is complete, the software allows you to see the volume where the cameras can determine the position of reflective markers. Figure 14 displays the capture volume used in this lab. The blue boxes show the capture volume, the blue pyramid shapes are the cameras and the black grid is the ground plane.

Figure 13. Camera View During Wanding

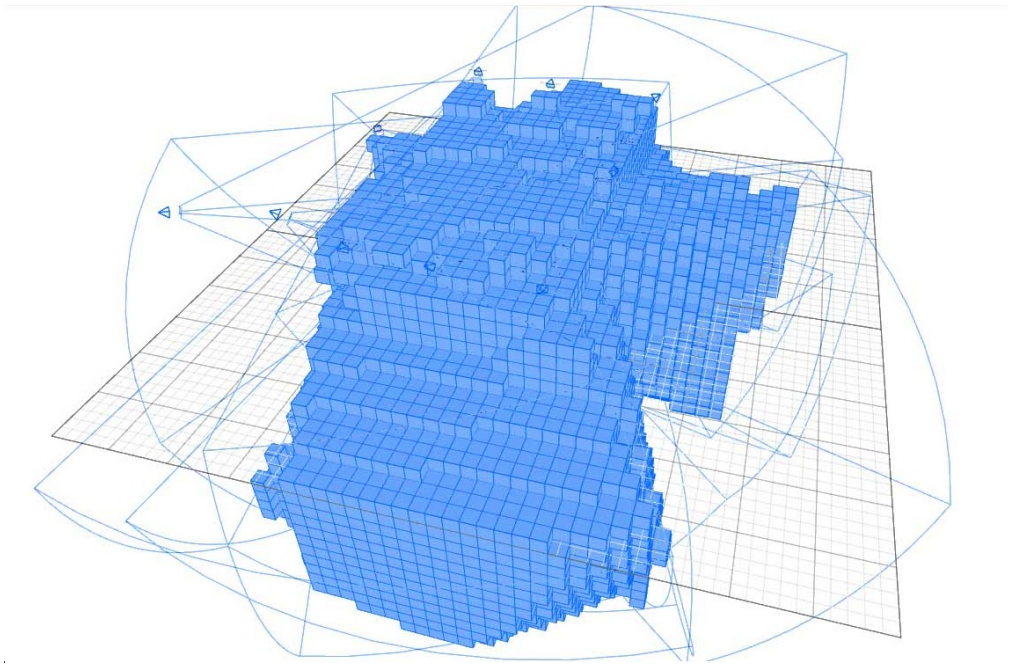

Figure 14. Capture Volume

4. Connectivity

All Optitrack cameras are controlled by the ground station computer. All cameras are connected to an Optitrack hub via USB 2.0 cables according to the OptiHub Setup guide shown in Figure 15. All hubs are interconnected via a synchronization cable and each hub is connected to the ground station computer via USB 2.0 cables. Up to five meters of extension cables may be used to connect the hubs to the ground station. Due to the size of the lab one extension cable was used for two of the hubs.

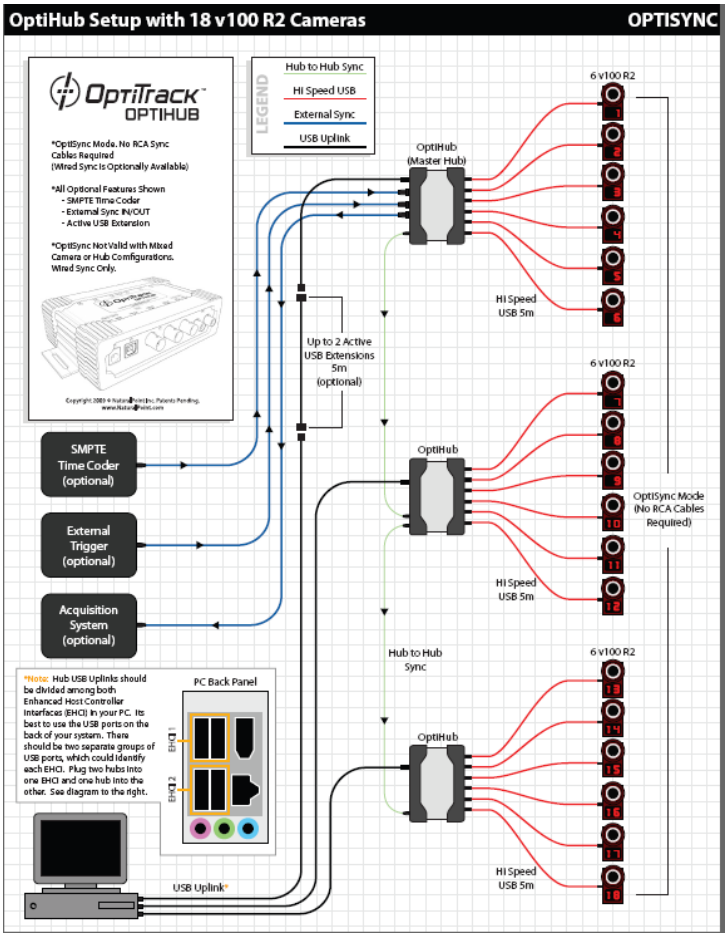

Figure 15. OptiHub Setup Guide

III. MODELING AND CONTROL OF THE QUADROTOR

A. INTRODUCTION

Before a trajectory generator can be built, one must first understand the dynamics of the vehicle. From the understanding of the dynamics of the vehicle, one can determine the aspects of a feasible trajectory as well as what controls need to be calculated to follow the trajectory. The dynamics of the quadrotor will be represented by a state space equation. The state space equation uses a set of matrices to set up a series of first-order differential equations of the vehicle states. Some flexibility exists in defining the matrix representing the vehicle state; called the state variable. The state variable can be selected through appropriate assumptions or approximations such as linearizing around a stable condition.

1. Assumptions

In order to simplify the complexity of the model, the following assumptions were made

- The Earth is flat and not rotating.
- The acceleration due to gravity is a constant 9.81 m/s.
- The quadrotor is symmetric about the pitch and roll axes and the moment of inertia about the x and y axes are equal.
- The quadrotor frame is a rigid body and does not flex.

- Pitch and roll angles are small.
- Vehicle speeds are slow enough that drag forces are negligible.

2. Coordinate Frames

In this paper, two coordinate frames are used; the body-fixed and local tangent plane. The body-fixed frame is fixed to the center of mass of the quadrotor and rotates with the vehicle. A Cartesian coordinate system is used with the x-direction pointing toward the front of the vehicle, the y-direction points to the left side of the vehicle, and the z-direction is upward. To determine angle directions, the right-hand-rule is used. For example, a positive roll angle will rotate the vehicle about the x-axis in a direction shown in Figure 16. An orange sticker is placed on the vehicle frame in the negative x-direction to mark the tail section and avoid confusion of vehicle orientation during flight. The local tangent plane (LTP) is also a Cartesian coordinate system that rotates with the earth. The LTP approximates the earth as a flat object and assumes that the earth is not moving. These approximations can be done because most quadrotor flights are very short in duration; hence they cannot travel far enough to experience neither the curvature of the planet nor the effects of the spinning earth. The LTP uses the x and y-directions in the horizontal plane and the positive z-direction is upward.

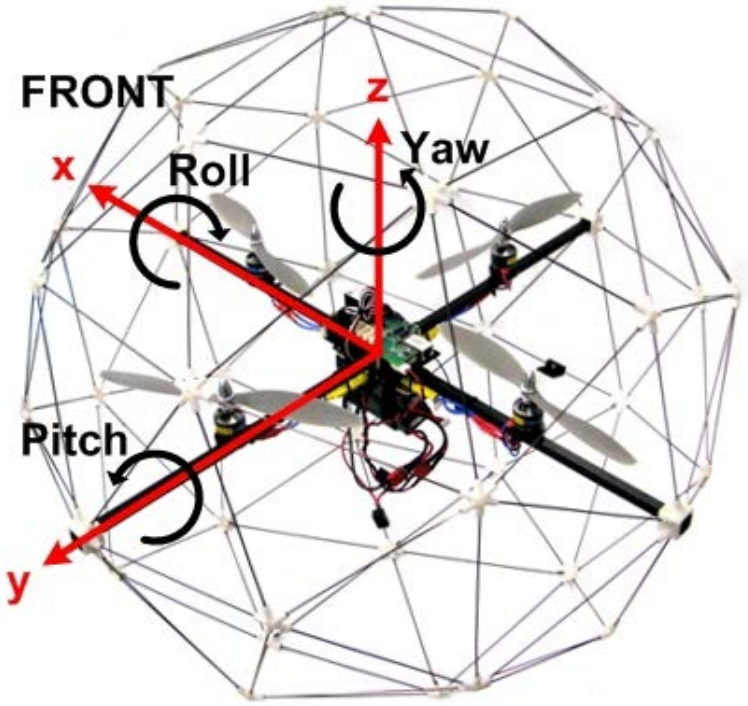

Figure 16. Quadrotor Body-Fixed Coordinate Frame

B. DYNAMICS

1. Actuator Dynamics

The quadrotor has four separate motor/propellers that can be independently controlled. Through control of these four motors, the second derivative of roll, pitch, yaw, and height in the body-fixed frame can be directly controlled. Through the control of these second derivatives, we can obtain control of position and yaw angle in the LTP frame.

The generated thrust from each propeller is modeled by the following first order system

$$F = K\frac{\omega}{s+\omega}u$$

where F is the generated force, K is a positive gain, ω is the actuator bandwidth, and u is the PWM input to the

motor. To simplify the model, a state variable, v will be used to represent the motor dynamics as

$v = \frac{\omega}{s+\omega} u$

For clarity, the motors on the quadrotor have been numbered in according to their location and direction of rotation. For the duration of this thesis, a number subscript will be used to denote each motor in accordance with Figure 17 [17].

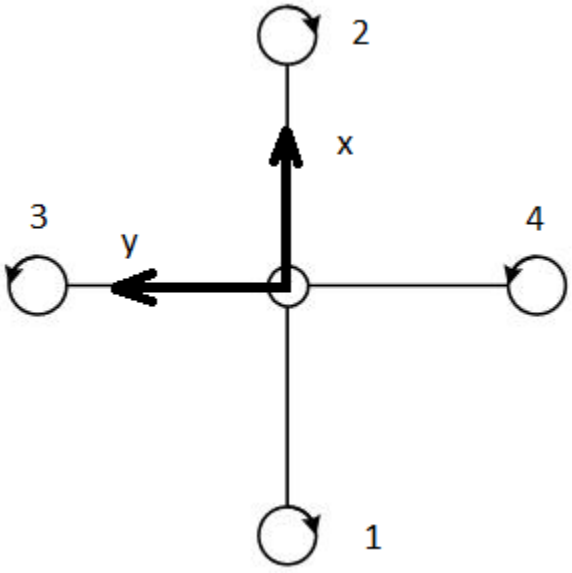

Figure 17. Motor Direction And Numbering

2. Pitch and Roll Model

a. Introduction

A pitch moment is created by increasing the lift created by motor number one and decreasing the lift created by motor number two by an equal amount. A roll moment can be created in a similar fashion by varying the lift force generated by motors three and four. The rotations caused by the pitch and roll moments will be about the center of mass

of the vehicle and rotations about each axis are assumed to be decoupled.

b. Model

For the roll and pitch model, we first need to establish some constants used in the model. A list of values determined by Quanser for the Qball-X4 can be found in Table 1. J is used as the mass moment of inertia of the quadrotor about the x and y axes. L is the distance from the motor to the center of gravity of the vehicle. $\Delta F_p = F_1 - F_2$ is the difference in lift force between the front and rear motors and $\Delta F_r = F_3 - F_4$ is the difference in lift force between the left and right motors. The roll and pitch rates denoted $\ddot{\phi}$ and $\ddot{\theta}$ respectively can be modeled as

$$J\ddot{\phi} = \Delta F_p L$$
$$J\ddot{\theta} = \Delta F_r L$$

Parameter	Value
K	120 N
ω	15 rad/s
J	0.03 kg m^2
M	1.4 kg
K_y	4 N m
J_{yaw}	0.04 kg m^2
L	0.2 m

Table 1. System Parameters [From 17]

If we use $\Delta u = u_1 - u_2$ and $\Delta u = u_3 - u_4$ in conjunction with previously developed formulas, we may put together the following state space equations for the pitch and roll model.

$$\begin{bmatrix} \dot{\phi} \\ \ddot{\phi} \\ \dot{v} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & \frac{KL}{J} \\ 0 & 0 & -\omega \end{bmatrix} \begin{bmatrix} \phi \\ \dot{\phi} \\ v \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ \omega \end{bmatrix} \Delta u_p \tag{1}$$

$$\begin{bmatrix} \dot{\theta} \\ \ddot{\theta} \\ \dot{v} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & \frac{KL}{J} \\ 0 & 0 & -\omega \end{bmatrix} \begin{bmatrix} \theta \\ \dot{\theta} \\ v \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ \omega \end{bmatrix} \Delta u_r \tag{2}$$

c. Control

The controller developed by Quanser uses the same setup for the pitch and roll controllers because it is assumed that the vehicle is symmetric about the two axes. The controller developed by Quanser is a linear quadratic regulator (LQR) that includes an integrator in the state variable. In order to include the integrator into the control law, equations (1) and (2) are appended as follows.

$$\begin{bmatrix} \dot{\phi} \\ \ddot{\phi} \\ \dot{v} \\ \dot{s} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & \frac{KL}{J} & 0 \\ 0 & 0 & -\omega & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \phi \\ \dot{\phi} \\ v \\ s \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ \omega \\ 0 \end{bmatrix} \Delta u_p \tag{3}$$

$$\begin{bmatrix} \dot{\theta} \\ \ddot{\theta} \\ \dot{v} \\ \dot{s} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & \frac{KL}{J} & 0 \\ 0 & 0 & -\omega & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \theta \\ \dot{\theta} \\ v \\ s \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ \omega \\ 0 \end{bmatrix} \Delta u_r \tag{4}$$

The LQR controller is designed assuming a state space model of the form $\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{B}u$ where **x** is the state variable, **A** and **B** are matrices determined by system dynamics, and u is the input to the system. The control law

is a feedback control of the form $u = -k\mathbf{x}$. The LQR controller uses a set of weighting matrices **Q** and **R** that are specified by the designer in order to minimize a set of costs [27]. These **Q** and **R** matrices in conjunction with the A and B matrices that describe the system, a simple Matlab command can be used to determine the gain, k to control the system.

The Q and R matrices developed by Quanser to be used for the pitch and roll controllers on the Qball-X4 are:

$$Q = \begin{bmatrix} 100 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 22000 & 0 \\ 0 & 0 & 0 & 10 \end{bmatrix} \quad (5)$$

$$R = 30000 \quad (6)$$

The feedback gain, k that results from these inputs results in poles at -19.827, -4.083+4.275i, -4.083–4.275i, and -0.316. In order to ensure that the quadrotor stays in the linear and stable region, limits have been placed at .2 radians for both pitch and roll angles.

3. Yaw Model

a. Model

The yaw model for a quadrotor is very simple in comparison to the other axes. The torque generated by each motor, thus the torque felt by the quadrotor by the air on the propellers is said to be proportional to the PWM input to the motors. If τ is the torque generated by the motor u is the PWM input to the motor, and K_y is a positive gain taken from Table 1, the following relationship is true:

$$\tau_i = K_y u_i \quad (7)$$

Rotation in the yaw axis is driven by a difference in torques from the motors rotating in opposite directions. If ψ is the yaw angle and $\Delta\tau = \tau_1 + \tau_2 - \tau_3 - \tau_4$, the yaw axis can be modeled as

$$J_y \ddot{\psi}_y = \Delta\tau$$

Written in state space form as:

$$\begin{bmatrix} \dot{\psi}_y \\ \ddot{\psi}_y \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \psi_y \\ \dot{\psi}_y \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{K_y}{J_y} \end{bmatrix} \Delta\tau \tag{8}$$

b. Control

The yaw controller for the Qball was designed in a similar fashion as the pitch and roll controllers. The Q and R matrices used are:

$$Q_{yaw} = \begin{bmatrix} 1 & 0 \\ 0 & .1 \end{bmatrix} \tag{9}$$

$$R = 1000 \tag{10}$$

The feedback gain, k that results from these inputs results in poles at -1.3532+1.1537i and -1.3532–1.1537i.

4. Position Model

a. Model

Motion in the horizontal plane, referred to here as the X-Y plane is due to thrust from the propellers when the vehicle has a non-zero pitch or roll angle. When the propeller thrust is not aligned with the vertical direction, a component of the thrust causes acceleration in the horizontal plane. The relationship between acceleration in the x and y directions, the average force from each

motor, mass, and the pitch and roll angles, denoted $\ddot{x}$, $\ddot{y}$, F, M, φ, and respectively is

$$M\ddot{x} = 4F\sin(\phi) \tag{11}$$

$$M\ddot{y} = -4F\sin(\theta) \tag{12}$$

If the pitch and roll angles are close to zero, the following linearization can be made

$$\begin{bmatrix} \dot{x} \\ \ddot{x} \\ \dot{v} \\ \dot{s} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & \frac{4K}{M}\theta & 0 \\ 0 & 0 & -\omega & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \\ v \\ s \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ \omega \\ 0 \end{bmatrix} u \tag{13}$$

$$\begin{bmatrix} \dot{y} \\ \ddot{y} \\ \dot{v} \\ \dot{s} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & -\frac{4K}{M}\phi & 0 \\ 0 & 0 & -\omega & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} y \\ \dot{y} \\ v \\ s \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ \omega \\ 0 \end{bmatrix} u \tag{14}$$

b. Control

An LQR controller was developed for the X-Y position model in the same manner as the pitch and roll controllers. The matrices used for the construction of the controller are

$$Q = \begin{bmatrix} 5 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & .1 \end{bmatrix} \tag{15}$$

$$R = 50 \tag{16}$$

The feedback gain, k that results from these inputs results in poles at -6.712, -1.61+0.792i, -1.61–0.792i, and -0.142.

5. Height Model

a. Model

Motion in the vertical direction is caused by the vertical component of the thrust from the propellers. If the thrust generated by each propeller is F, g is the acceleration due to gravity, and $\ddot{z}$ is the vertical acceleration, then

$$M\ddot{z} = 4F\cos(\phi)\cos(\theta) - Mg$$

The vertical channel can be modeled by the following state space formula

$$\begin{bmatrix} \dot{z} \\ \ddot{z} \\ \dot{v} \\ \dot{s} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & \frac{4K}{M} & 0 \\ 0 & 0 & -\omega & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} z \\ \dot{z} \\ v \\ s \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ \omega \\ 0 \end{bmatrix} u - \begin{bmatrix} 0 \\ g \\ 0 \\ 0 \end{bmatrix} \quad (17)$$

b. Control

Unlike the other controllers, the vertical channel is controlled via a proportional integral derivative (PID) controller. A PID controller uses three gains and information about the state, its derivative and the integral of the state. The gains used on for the height controller for the Qball-X4 are given in Table 2.

Gain	Designation	Value
Proportional	K_p	0.00621
Integral	K_i	0.0015
Derivative	K_d	0.0078

Table 2. Height Controller Gains

C. MODES OF CONTROL

The model developed by Quanser allows the user to easily select closed loop control for altitude, yaw angle, and horizontal position. Also, if the user wants to take manual control, the model can be configured on the fly to take commands from a joystick connected to the ground station computer. The commands from the joystick are fed through the host model along with the vehicle position from the Optitrack system over the wireless connection.

THIS PAGE INTENTIONALLY LEFT BLANK

IV. DIRECT METHOD BASED TRAJECTORY GENERATION

A. INTRODUCTION

In order to achieve full autonomy, we need to develop a optimal trajectory generator that can spatially de-conflict paths as well as respect vehicle constraints. The routine must be capable of updating the path several times throughout the flight in order to account for any discrepancies in the model as well as disturbances in the process. Most of the known optimization software packages such as OTIS, SOCS, DIRCOL, or DIDO, based on collocation, direct transcription and pseudo spectral methods [21]-[14], require extensive computational power as they involve many varied parameters and therefore may not have the ability to be used for on-line computation of agile short-term maneuvers.

The method proposed to use by this thesis is the direct method of calculus of variations exploiting the inverse dynamics of a vehicle in a virtual domain (IDVD) [27]. This method is capable of fast prototyping of optimal trajectories for multiple vehicles. It allows respecting vehicle constraints and assures collision-free trajectories. By design IDVD method involves very few varied parameters and significantly reduced computational power [27]. This method is also useful in that it is relatively simple to modify and use, giving the user the ability to modify the code for a specific scenario or change the number of vehicles operating together.

The trajectory generator gives a set of x, y, and z coordinates and corresponding time derivatives. These

coordinates can be fed directly to the quadrotor, or the vehicle's inverse dynamics can be used to compute the necessary controls to track the reference trajectory.

B. REFERENCE TRAJECTORY

In order to create a reference trajectory independent of any time derivative constraints, a path is created using a mathematical function for each Cartesian coordinate using a virtual variable "τ" as the independent variable. Since this reference function is in the virtual domain, we have completely decoupled space and time. Later, we will introduce a variable speed factor to map the function from the virtual domain to the time domain [25].

When selecting a reference function (set of basis functions), we must consider the general shape for an expected trajectory as well as the number of boundary conditions that must be satisfied. In the case of a quadrotor, a simple polynomial will give the desired shape for the reference function, but other shapes such as a sinusoid may be selected based on the operating environment. We represent the x, y, and z coordinates as analytically defined *N*th-order polynomials of the form

$$x = \sum_{k=0}^{N} a_{xk} \tau^k \tag{18}$$

$$y = \sum_{k=0}^{N} a_{yk} \tau^k \tag{19}$$

$$z = \sum_{k=0}^{N} a_{zk} \tau^k \tag{20}$$

The order of the reference function polynomial N is determined by the number of boundary conditions that must be satisfied, but can be increased to add to the degrees of

freedom. If d_0 is the highest-order spatial derivative of the set of initial conditions and d_f is the highest order derivative of the final conditions, then the order of polynomial used, is $N = d_o + d_f + 1$. For example, if we desired to specify third order derivatives at the initial and final points, then $d_o = 3$, $d_f = 3$, and $N = 3+3+1 = 7$.

To construct the reference trajectory, we must ensure that all initial and final conditions are satisfied. In order to ensure that the polynomial satisfies the velocity and acceleration at the endpoints of the maneuver, we must analytically compute $\frac{dx}{d\tau}$, $\frac{dy}{d\tau}$, $\frac{dz}{d\tau}$, and higher order derivatives. Since we will be computing coefficients of higher order derivatives later, it is convenient to rewrite equations (18)-(20) and their derivatives as

$$x = \sum_{k=0}^{N} a_{xk} \tau^k \frac{(\max(1, k-2))!}{k!} \tag{21}$$

$$y = \sum_{k=0}^{N} a_{yk} \tau^k \frac{(\max(1, k-2))!}{k!} \tag{22}$$

$$z = \sum_{k=0}^{N} a_{zk} \tau^k \frac{(\max(1, k-2))!}{k!} \tag{23}$$

$$\frac{dx}{d\tau} = x' = \sum_{k=1}^{N} a_{xk} \tau^{k-1} \frac{(\max(1, k-2))!}{(k-1)!} \tag{24}$$

$$\frac{dy}{d\tau} = y' = \sum_{k=1}^{N} a_{yk} \tau^{k-1} \frac{(\max(1, k-2))!}{(k-1)!} \tag{25}$$

$$\frac{dz}{d\tau} = z' = \sum_{k=1}^{N} a_{zk} \tau^{k-1} \frac{(\max(1, k-2))!}{(k-1)!} \tag{26}$$

$$\frac{d^2x}{d\tau^2} = x'' = \sum_{k=2}^{N} a_{xk} \tau^{k-2} \tag{27}$$

$$\frac{d^2y}{d\tau^2} = y'' = \sum_{k=2}^{N} a_{yk} \tau^{k-2} \tag{28}$$

$$\frac{d^2z}{d\tau^2} = z'' = \sum_{k=2}^{N} a_{zk} \tau^{k-2} \tag{29}$$

$$\frac{d^3x}{d\tau^3} = x''' = \sum_{k=3}^{N}(k-2)a_{xk}\tau^{k-3} \tag{30}$$

$$\frac{d^3y}{d\tau^3} = y''' = \sum_{k=3}^{N}(k-2)a_{yk}\tau^{k-3} \tag{31}$$

$$\frac{d^3z}{d\tau^3} = z''' = \sum_{k=3}^{N}(k-2)a_{zk}\tau^{k-3} \tag{32}$$

For this thesis, it was important ensure the vehicle experienced a smooth transition at the beginning and endpoints of the trajectory. It is also necessary to allow for another set of varied parameters to allow for the algorithm to have flexibility over the shape of the trajectory. Since the initial and final position, velocity, and accelerations are already specified, the third order derivative called the jerk will be used as the varied parameter. Now we have four initial and four final conditions to be satisfied, calling for a 7th order polynomial of the form expressed in equations (21)-(32). If we select the final τ as a variable parameter and combine equations (21)-(32), we can setup a matrix equation to solve for all polynomial coefficients a_{x0} through a_{x7}.

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & \tau_f & \frac{1}{2}\tau_f^2 & \frac{1}{6}\tau_f^3 & \frac{1}{12}\tau_f^4 & \frac{1}{20}\tau_f^5 & \frac{1}{30}\tau_f^6 & \frac{1}{42}\tau_f^7 \\ 0 & 1 & \tau_f & \frac{1}{2}\tau_f^2 & \frac{1}{3}\tau_f^3 & \frac{1}{4}\tau_f^4 & \frac{1}{5}\tau_f^5 & \frac{1}{6}\tau_f^6 \\ 0 & 0 & 1 & \tau_f & \tau_f^2 & \tau_f^3 & \tau_f^4 & \tau_f^5 \\ 0 & 0 & 0 & 1 & 2\tau_f & 3\tau_f^2 & 4\tau_f^3 & 5\tau_f^4 \end{bmatrix} * \begin{bmatrix} a_{x0} \\ a_{x1} \\ a_{x2} \\ a_{x3} \\ a_{x4} \\ a_{x5} \\ a_{x6} \\ a_{x7} \end{bmatrix} = \begin{bmatrix} x_0 \\ x_0' \\ x_0'' \\ x_0''' \\ x_f \\ x_f' \\ x_f'' \\ x_f''' \end{bmatrix} \tag{33}$$

Matrix equations for the two other coordinates are similar to (33) and omitted here.

Now that we have developed a method for determining the reference function, it is useful to explore the versatility of the function. Figure 18 shows the effects of varying the initial third order derivative and final value for tau. The Figure was developed using the same algorithm inside the trajectory generator used for this thesis.

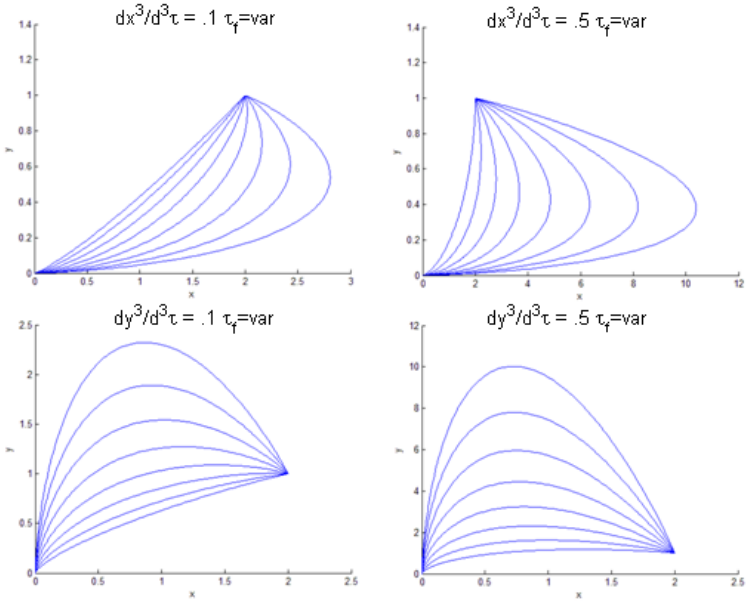

Figure 18. Illustration of Reference Function Flexibility Using Two Varied Parameters

C. SPEED FACTOR

As previously stated, in order to decouple space and time, a speed factor was introduced. The speed factor, λ links the virtual domain to the time domain by the following mapping function

$$\lambda = \frac{d\tau}{dt} \tag{34}$$

In order to allow for flexibility in the trajectory and simplicity of the model, a polynomial was selected to

represent $\lambda(\tau)$. At this point it is convenient to visit the matter of linking time derivatives of $x(\tau)$and x(t). The boundary conditions used in calculating the coefficients of the reference functions were derivatives of τ whereas we only know the boundary conditions as derivatives with respect to time. In order to relate these boundary conditions, we simply set the boundary conditions of λ at unity so that $d\lambda = dt$ and all virtual derivatives are equal to corresponding timer derivatives. To increase the flexibility of the speed function, the 2nd order derivatives at the endpoints will be used as varied parameters and the 1st order time derivatives will be set at zero, giving us six boundary conditions to satisfy and requiring a 5th order polynomial. If N=5, the polynomial and derivatives of lambda is represented as [26]

$$\lambda = \sum_{k=0}^{N} a_{\lambda k} \tau^k \frac{(\max(1, k-2))!}{k!} \tag{35}$$

$$\frac{d\lambda}{d\tau} = \lambda' = \sum_{k=1}^{N} a_{\lambda k} \tau^{k-1} \frac{(\max(1, k-2))!}{(k-1)!} \tag{36}$$

$$\frac{d^2\lambda}{d\tau^2} = \lambda'' = \sum_{k=2}^{N} a_{\lambda k} \tau^{k-2} \tag{37}$$

If we want to satisfy all boundary conditions and equations (34)-(35) then we can setup the following matrix equation to solve for the speed factor polynomial coefficients where $\lambda_0 = \lambda_f = 1$ $\lambda_0' = \lambda_0' = 0$ and λ_0'' and λ_f'' are varied parameters.

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & \tau_f & \frac{1}{2}\tau_f^2 & \frac{1}{6}\tau_f^3 & \frac{1}{12}\tau_f^4 & \frac{1}{20}\tau_f^5 \\ 0 & 1 & \tau_f & \frac{1}{2}\tau_f^2 & \frac{1}{3}\tau_f^3 & \frac{1}{4}\tau_f^4 \\ 0 & 0 & 1 & \tau_f & \tau_f^2 & \tau_f^3 \end{bmatrix} * \begin{bmatrix} a_{\lambda 0} \\ a_{\lambda 1} \\ a_{\lambda 2} \\ a_{\lambda 3} \\ a_{\lambda 4} \\ a_{\lambda 5} \end{bmatrix} = \begin{bmatrix} \lambda_0 \\ \lambda_0' \\ \lambda_0'' \\ \lambda_f \\ \lambda_f' \\ \lambda_f'' \end{bmatrix} \tag{38}$$

D. INVERSE DYNAMICS

To determine the controls that need to be fed to the vehicle, the quadrotors inverse dynamics need to be determined. The quadrotor uses roll and pitch angles in coordination with a yaw angle to control position in the horizontal plane and propeller thrust to control vertical height. Since roll, pitch, yaw, and thrust are used to control the vehicle, we must develop a set of methods to calculate these values to be fed to the vehicle in order to track the trajectory. From the geometry of the quadrotors dynamics, we can determine that the pitch and roll angles and thrust are [27]

$$\theta = \arctan\left(\frac{\ddot{x}}{g - \ddot{z}}\right) \tag{39}$$

$$\phi = \arcsin\left(\frac{-\ddot{y}}{\sqrt{\ddot{x}^2 - \ddot{y}^2 - \left(g - \ddot{z}\right)^2}}\right) \tag{40}$$

$$F = \frac{(\ddot{z} - g)}{\cos(\theta)\cos(\phi)} \tag{41}$$

To compute the time derivative, the speed factor and virtual derivatives of the reference function were utilized in the following manner (here ζ represents any state) [27].

$$\frac{d\zeta}{dt} = \dot{\zeta} = \lambda \frac{d\zeta}{d\tau} \tag{42}$$

$$\frac{d^2\zeta}{dt^2} = \ddot{\zeta} = \lambda\left(\frac{d\lambda}{d\tau}\frac{d\zeta}{d\tau} + \lambda\frac{d^2\zeta}{d\tau^2}\right) \tag{23}$$

The spatial trajectory is given in three axes, thus only three independent control parameters need to be varied in order to track the trajectory assuming the orientation of the vehicle is not a concern. Since ψ effectively only controls the orientation of the quadrotor, it is kept at zero. Later, this could be varied in order to aid in the accomplishment of mission objectives such as reconnaissance or tracking other vehicles with onboard sensors.

E. COST FUNCTION

1. Single Quadrotor Trajectories

Now that we have created a trajectory that satisfies the boundary conditions, it is time to optimize it. In order to optimize the trajectory, we must develop a cost function. For the case of one quadrotor, we are concerned with two factors; deviation in arrival time and respecting the vehicle constraints. The vehicle constraints that we are concerned about are the pitch and roll angles that will be determined by accelerations in the horizontal plane. Limits in the vertical direction are not considered here as deviations in the vertical direction are usually quite small. From these constraints, the cost function, J was constructed as

$$J = C_1 \frac{\left(t_{desired} - t_{end}\right)^2}{t_{desired}^2} + C_2 10^{\phi_{\max}/\phi_{threshold}} C_3 10^{\theta_{\max}/\theta_{threshold}} \quad (44)$$

Where C_1, C_2, and C_3 are constants to be tuned later, $t_{desired}$, t_{end}, φ_{max}, and $\varphi_{threshold}$ are the desired time entered by the user, end time of the maneuver, maximum pitch angle in the maneuver, maximum pitch angle the controller will allow respectively. The general shape of each portion of

the cost function is parabolic and minimized when $t_{desired} = t_{end}$, φ=0, and θ=0. Clearly we do not want the pitch and roll angles to be zero, otherwise the vehicle will not move in the horizontal plane. The cost function (35) will allow for some deviation of the roll angles from the horizontal position, but penalizes them more as they approach the threshold values for the controller.

2. Multiple Quadrotor Trajectories

In the real-world situation, each vehicle is supposed to carry its own "see and avoid" hardware, so its computer will only be responsible for computation of its own collision-avoiding maneuver. In this thesis quadrotors were not supposed to possess a "see" capability, so that the trajectories for both vehicles were computer in a centralized manner. It means that if we have M vehicles, the individual number of varied parameters for the optimization problem has to be multiplied by M. In order to mitigate the increase of varied parameters this thesis assumed a certain symmetry for multiple vehicles which allowed keeping the number of varied parameters at the same level as for a single vehicle. Specifically, each vehicle uses the same speed factor and the third order timer derivatives are in opposite directions at the boundary points.

a. Simplifications

For this thesis, the trajectory generator was created for controlling only two vehicles at one time. Using only two vehicles allows the testing to take place in the small laboratory that is available while still proving

that multiple vehicles can be controlled in this manner. To simplify the problem and decrease the number of varied parameters, the same speed factor was used for both vehicles. Using the same speed factor still allows for the de-coupling of space and time, yet decreases the need for extra varied parameters. To further decrease the number of varied parameters, a relationship was created between the initial and final jerk for each vehicle that ensured the two sets of trajectories diverged from one another. The relationship used was

$$\frac{d^3x_A}{dt^3} = -\frac{d^3x_B}{dt^3} \tag{45}$$

$$\frac{d^3y_A}{dt^3} = -\frac{d^3y_B}{dt^3} \tag{46}$$

Now that we have a set of equations relating the third order boundary conditions of each vehicle in the horizontal plane, we only need to vary the parameters for one vehicle, which will decrease computational time. Finally, we often do not want the trajectory of the vehicles to take them underneath each other because of interference with the sonar sensors that are used to determine altitude. It is also deemed unnecessary to allow for extra freedom of motion in the vertical channel, so the jerk in the vertical direction was not used as a varied parameter and consequently set to zero. The list of assumptions used here reduces the number of varied parameters from 18 down to seven.

b. Modification of the Cost Function

To incorporate the extra vehicle, the cost function was modified to include the extra set of Euler

angles, arrival time, as well as an extra term for the distance between the vehicles

$$J = C_1 \frac{\left(t_{desired} - t_{1end}\right)^2}{t_{desired}^2} + C_2\left(t_{2end} - t_{1end} - \Delta T\right)^2 + C_3\left(d_{\min} - d_{threshold}\right)^2 + \sum_{i=1}^{2}\left(C_{2i+2}10^{\frac{\phi_{i\max}}{\phi_{threshold}}} + C_{2i+3}10^{\frac{\theta_{i\max}}{\theta_{threshold}}}\right) \quad (47)$$

where d_{min} is the minimum distance between the vehicles along the trajectory and $d_{threshold}$ is 1.2m.

As seen the Euler angles penalties are identical for both vehicles. The arrival time term for the second vehicle is essentially the same as for the first vehicle if $\Delta T = 0$. Written slightly different it allows sequencing the arrival time for multiple vehicles in the future research if needed.

The distance portion of the cost function (38) was created so that the minimum distance between the vehicles would be driven to a pre-determined value that ensured the vehicles did not collide. The value that was used in this thesis was 1.2m, which corresponds to a distance slightly greater than twice the radius to allow for a safety margin.

c. Computing the Final Trajectory

In order to compute a trajectory that satisfies all constraints, a Simulink model is created that solves equations (33) and (38) and creates the trajectory as a function of time. From this trajectory, the arrival time, all vehicle controls, and the relative distance between vehicles can be calculated for every point along the trajectory. From this information calculated by the

Simulink model, we can determine the value of the cost function. The fminsearch command is used inside Matlab to run the Simulink model and determine a set of varied parameters that minimizes the cost function, hence, an optimal trajectory. The Simulink model was set to discretize the model into 200 equally spaced fixed-steps. Since the output of the function would not match the frequency of the controller, the Matlab command “interp1” was used to perform a linear interpolation between the points at the controller frequency of 200 Hz. The final set of varied parameters used with the minimization function were $\left.\frac{d^3x}{dt^3}\right|_o$ $\left.\frac{d^3y}{dt^3}\right|_o$ $\left.\frac{d^3x}{dt^3}\right|_f$ $\left.\frac{d^3y}{dt^3}\right|_f$ $\left.\frac{d^2\lambda}{d\tau^2}\right|_o$ $\left.\frac{d^2\lambda}{d\tau^2}\right|_f$ and τ_f. Once the trajectory has been computed, either the Euler angles or position commands can be fed to the quadrotor controllers. The optimization routine will be continuously run during the flight to allow for changes in the mission and account for disturbances or inaccuracies in the model.

V. ILLUSTRATIVE MISSIONS AND SIMULATION RESULTS

1. INTRODUCTION

The trajectory generator was verified using two representative scenarios that were specifically designed to test for collision avoidance. By examining the outputs of the trajectory, we were able to confirm that the vehicle constraints and arrival time were satisfied.

2. APPROACH

Because of time constraints and the short duration of maneuvers, the trajectory for each quadrotor was only computed once on the ground station computer and fed to the quadrotors via the wireless connection. At the beginning of each flight, a waypoint was fed to each controller for 20 seconds after take-off to allow for the transient response to settle out. Following the initial 20 seconds, a set of Euler angles was fed to the pitch and roll controller and an altitude command was fed to the altitude controller followed by a second waypoint at the endpoint of the maneuver. Since the Euler angles generated by the trajectory were small, the results from this method were unreliable. When the model is started, the Euler angles are initialized at zero and then the gyroscope is used to update the angles during flight. With this method of initialization, if the initial orientation was not perfectly level, there will always be a steady state error in the estimation of the orientation of the vehicle. For this reason, it was decided to use the position data from the trajectory to control the quadrotors. During each flight, there was some lag in the position of the

quadrotor. After observing the response of the Qball to position commands and examining the controller, it is believed that the native controller developed by Quanser is not optimal and needs further work to increase the response time.

All calculations were performed on a desktop PC with an Intel Core i7 2.79 GHz processor and 8GB of RAM running Matlab version 7.10.0.

3. SCENARIO #1

The first scenario that was tested was a situation where each quadrotor started at one edge of the room from rest (zero velocity and acceleration in all directions) and the vehicles were told to swap places, again with zero velocity and acceleration. The arrival time for the maneuver was requested to be 20 seconds. This scenario was selected to test the ability of the trajectory generator to create a path that avoided collisions. If a simple set of waypoints were given to the quadrotors, they would drive toward each other and collide. Table 3 gives a list of initial and final conditions for each vehicle that was fed into the trajectory generator.

	Quadrotor A	Quadrotor B
x_0	-1m	2m
y_0	-0.5m	-0.5m
z_0	.6m	.6m
x_f	2m	-1m
y_f	-0.5m	-0.5m
z_f	.6m	.6m
$\dot{x}_0$	0	0
$\dot{y}_0$	0	0

$\dot{z}_0$	0	0
$\dot{x}_f$	0	0
$\dot{y}_f$	0	0
$\dot{z}_f$	0	0
$\ddot{x}_0$	0	0
$\ddot{y}_0$	0	0
$\ddot{z}_0$	0	0
$\ddot{x}_f$	0	0
$\ddot{y}_f$	0	0
$\ddot{z}_f$	0	0

Table 3. Scenario #1 Boundary Conditions

The CPU time required to create the trajectory was 25.5374 seconds for a single run. The final values for each varied parameter can be found in Table 4 and a bird-eye view of the trajectory can be seen in Figure 19.

Varied Parameter	Value
τ_f	16.04
$\dddot{x}_i$	0.0883
$\dddot{y}_i$	0.5489
$\dddot{x}_f$	0.0883
$\dddot{y}_i$	-0.0549
λ_i''	-0.0440
λ_f''	-0.0440

Table 4. Varied Parameters for Scenario #1

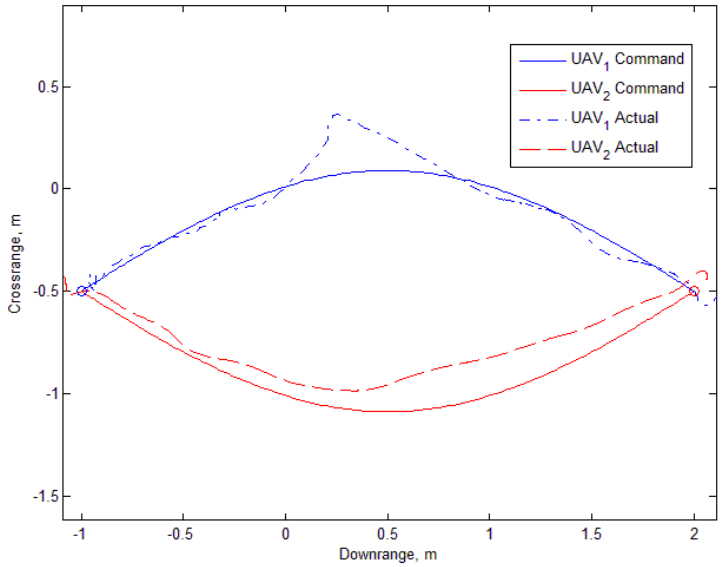

Figure 19. Scenario #1 Reference Trajectory and Actual Data

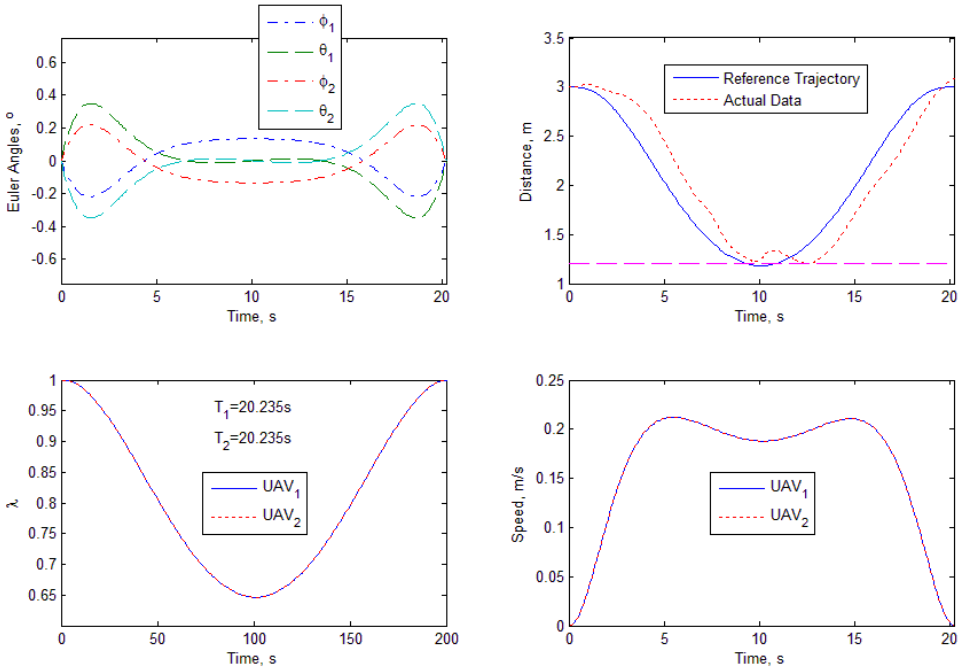

Figure 20. Scenario #1 Parameters

The trajectory generator develops a feasible path for this scenario that mimics the path that a human operator would fly. The algorithm performs well here because the trajectory selected is the shortest path that still allows for a 1.2m separation between the two vehicles. A quick look at Figure 19 shows the lack of performance of the native controller. UAV_1 shown in blue was pushed of its trajectory due to wind turbulence caused by the two vehicles. This disturbance seemed to reach a maximum at the midpoint of the maneuver because it was closest to the edge of the room where the airflow pattern changes. The inability of the controller to track commands is displayed by the performance of UAV_2. The position of the vehicle is always inside the trajectory path, illustrating the need for a safety margin when specifying the minimum threshold distance between the vehicles.

4. SCENARIO #2

The second scenario tested the ability of the algorithm to create a trajectory to cross the paths of both vehicles. Again, both vehicles are starting from rest at a distance of 1.5m from each other on one side of the lab. The final desired position of each vehicle fed into the trajectory generator was at the other edge of the lab, but the quadrotors must switch positions in one coordinate before they reach the other edge and the desired time of the maneuver was 20 seconds. Again, this scenario tests the collisions avoidance aspect of the algorithm as well as the arrival time of the vehicles.

	Quadrotor A	Quadrotor B
x_0	-0.75m	-0.75m
y_0	-1.25m	0.25m
z_0	.6m	.6m
x_f	1.75m	1.75m
y_f	0.25m	-1.25m
z_f	.6m	.6m
$\dot{x}_0$	0	0
$\dot{y}_0$	0	0
$\dot{z}_0$	0	0
$\dot{x}_f$	0	0
$\dot{y}_f$	0	0
$\dot{z}_f$	0	0
$\ddot{x}_0$	0	0
$\ddot{y}_0$	0	0
$\ddot{z}_0$	0	0
$\ddot{x}_f$	0	0
$\ddot{y}_f$	0	0
$\ddot{z}_f$	0	0

Table 5. Scenario #2 Boundary Conditions

The time required to compute this trajectory was slightly higher than scenario #1 at 45.1935 seconds. The varied parameters and trajectory that was calculated for scenario #2 can be seen in Table 6 and Figures 21 and 22.

Varied Parameter	Value
τ_f	16.04
$\ddot{x}_i$	0.0883
$\ddot{y}_i$	0.5489
$\ddot{x}_f$	0.0883
$\ddot{y}_i$	-0.0549
λ_i''	-0.0440
λ_f''	-0.0440

Table 6. Varied Parameters for Scenario #2

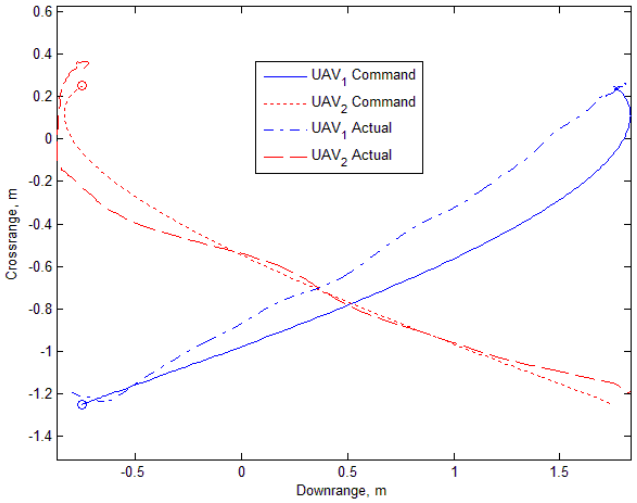

Figure 21. Scenario #2 Reference Trajectory and Actual Data

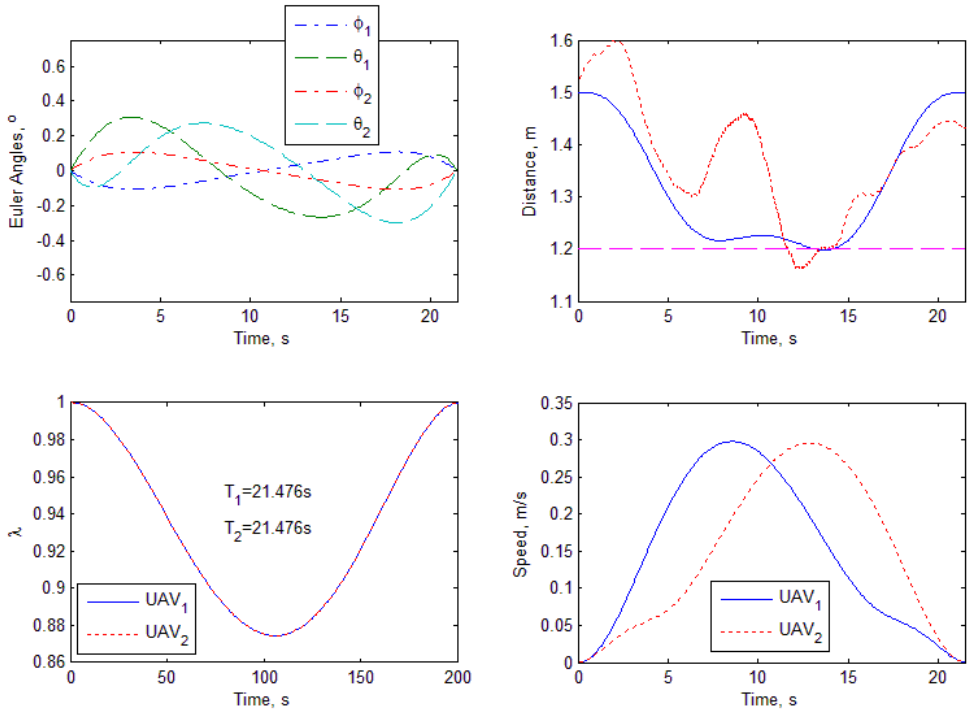

Figure 22. Scenario #2 Parameters

In scenario #2, the algorithm develops a trajectory that is not necessarily intuitive, but does satisfy all requirements. The initial jerk of UAV_2 allows the other quadrotor to cross ahead and avoid a collision at the geographic center of the maneuver. A look at Figure 22 shows that the minimum distance between the vehicles is 1.2m at the 14 second mark. Figure 22 shows the lack of performance of the controller that was also seen in Figure 19. It is important to note that there was an error in the arrival time of the trajectory of 1.476 seconds. Although this represents a small deviation in this short maneuver, it may represent a larger deviation in a maneuver in a larger scenario. The deviation in arrival time could be solved by giving the user the ability to adjust the gains in the cost matrix. If arrival time is more important to

the user, then he can select a larger gain for C_1 and C_2, the gains for the arrival time terms. A deviation in arrival time in this instance is also likely due to the poor performance of the position controller, and will have to be reevaluated once a more accurate controller is developed.

THIS PAGE INTENTIONALLY LEFT BLANK

VI. CONCLUSIONS AND FUTURE WORK

1. CONCLUSIONS

The following conclusions can be drawn from the research conducted within this thesis:

- We have developed an architecture that allows the control of multiple UAVs from a single ground station.
- The inverse dynamics in the virtual domain based trajectory generator has proved to be a good candidate for a “see and avoid” capability for future unmanned systems.
- The native controller developed by Quanser does not exhibit an acceptable behavior and needs to be optimized for better performance while tracking spatial curved trajectories.

2. RECCOMENDATIONS FOR FUTURE RESEARCH

Recommendations for future work with the Qball-X4 quadrotor as a result of this thesis are:

- Investigate latency issues in controlling multiple unmanned systems.
- Improve the trajectory tracking controller of the Qball-X4 by either optimizing the native LQR controller or developing a model predictive controller.

- Incorporate more than two quadrotors or multiple heterogeneous vehicles to test the scalability of the system.
- Add sensors to give each vehicle the ability to sense the position of other cooperative vehicles.
- Incorporate compiled C++ code to enable reactive trajectory updates onboard each vehicle.
- Decrease computational load to allow more rapid trajectory updates to enable a real-time system.
- Reduce simplifying symmetry assumptions to result in more practical and intuitive solutions.

APPENDIX

MATLAB DOCUMENTATION

filter_design.m file

This script is run when the Qball-X4 model is initialized in order to calculate the coefficients for complimentary filters.

```
t=10;
s = tf('s');
Gg = t^2*s/(t*s+1)^2
Gi = (2*t*s+1)/(t*s+1)^2
```

controller_design.m file

This script is run when the Qball-X4 model is initialized and calculates the controller parameters.

```
% PITCH and ROLL
wnfom = 15;
L = 0.2;
w = 15;
K = 120;
J = 0.03;
Jyaw = 0.04;
CLimit = 0.025;
M = 1.4;
g = 9.8;

Am = [0 1 0
    0 0 K*L/J
    0 0 -w];
Bm = [0 0 w]';
Aobs = Am' ;
Bobs = eye(3);
Qobs = diag([.001 10000 .01]);

Robs = diag([ 1 1 1 ])*1;
Kobs = lqr(Aobs,Bobs,Qobs,Robs)
Kobs = Kobs';
Aobs = Aobs'-Kobs*Bobs';
eig(Aobs)
Bobs = [Bm Kobs]
Cobs = eye(3)
Dobs = [ 0 0 0 0
    0 0 0 0
    0 0 0 0];
```

```
% augment with integrator
Ai = [Am [0 0 0 ]'
      1 0 0 0 ];
Bi = [Bm' 0]';
Ci = eye(4);
Di = [0 0 0 0 ]';
Q = diag([100 0 22000 10]);
R = 30000;

ki = lqr(Ai,Bi,Q,R);
rp_eig = eig(Ai-Bi*ki);
fprintf ('********************************************** \n');
fprintf('ROLL, PITCH DESIGN \n');
fprintf( 'P = %5.3f D = %5.3f Actuator = %5.3f I = %5.3f \n\n',ki(1),
ki(2),ki(3),ki(4));
for i = 1:4
fprintf('  %5.3f + %5.3f i \n  ',real(rp_eig(i)), imag(rp_eig(i)));
end;


%POSITION CONTROLLER (C2)
% XZ travel
xyposition = 1
tlimit = 5*pi/180; %max pitch cmd radians
%tlimit = 15*pi/180; %max pitch cmd radians
vlimit = 0.3; % max speed cmd in m/sec
%vlimit = 0.5; % max speed cmd in m/sec
Tau_theta = 1/7; % closed loop time constant for pitch response
wt =1/Tau_theta; %closed loop theta bandwidth
kt = 1;
a = [0 1 0 0
    0 0 g 0
    0 0 -wt 0
    1 0 0 0 ];
b = [0 0 wt 0 ]';

q = diag([ 5 2 0 0.1]);
r = 50;

k = lqr(a,b,q,r);

ac = a-b*k;
xy_eig = eig(a-b*k);
Kp = k(1);
Kd = k(2);
Ki = k(4);
Kw = k(3);
fprintf('\n\n X Y Design \n');
fprintf( 'P = %5.3f D = %5.3f Actuator = %5.3f I = %5.3f \n\n',k(1),
k(2),k(3),k(4));
for i = 1:4
fprintf('  %5.3f + %5.3f i \n  ',real(xy_eig(i)), imag(xy_eig(i)));
end;
```

```
% Z axis without actuator

vlimith = 0.1;
Amh = [0 1
    0 0 ]
Bmh = [0 4*K/M]';
Cmh = [1 0];
Dmh = 0;

% augment with integrator
Aih = [Amh [0  0 ]'
      1  0 0 ];
Bih = [Bmh' 0]';
Cih = eye(3);
Dih = [0 0 0]';

Q = diag([1 0 50]);
R = 5000000;
kh = lqr(Aih,Bih,Q,R);
h_eig = eig(Aih-Bih*kh);
fprintf ('********************************************** \n');
fprintf('Z DESIGN \n');
fprintf( 'P = %5.3f D = %5.3f I = %5.3f \n\n',kh(1), kh(2),kh(3));
for i = 1:3
fprintf('  %5.3f + %5.3f i \n  ',real(h_eig(i)), imag(h_eig(i)));
end;
Kph = kh(1);
Kdh = kh(2);
Kwh = 0;
Kih = kh(3);

% yaw axis
yaw = 1
Ky = 4;
Jy = 0.04;

Amy = [0 1
    0 0 ];
Bmy = [0 Ky/Jy]';
Cmy = eye(2);
Dmy = [0;0];

Qy = diag([1 0.1]);
Ry = 1000;
ky = lqr(Amy,Bmy,Qy,Ry)
h_eigy = eig(Amy-Bmy*ky)
Kpyaw = ky(1)
Kdyaw = ky(2)



Bih = [Bih,[0 1 0]'];
Dih = [Dih, [0 0 0]'];
```

DMlopt.m file

This script uses the fminsearch and DMlfun function to run the Simulink model to minimize the discrepancy function.

```
%% Initial guesses for varied parameters
x0=[20/1000      % tauf
       0.08      % X0_tpl_prime
       5/10      % dirX0_tpl_prime
       0.08      % Xf_tpl_prime
      -5/10      % dirXf_tpl_prime
      -0.02      % lam0_2pr
      -0.02];    % lamf_2pr

%% Optimization
t = cputime;
options=optimset('TolFun', 1e-1, 'TolX', 1e-1, 'Display', 'final');
[x0,fval,exitflag]=fminsearch(@DMlfun,x0)
time_elapsed = cputime-t

%% Optimal values of varied parameters
tauf            = x0(1);
X0_tpl_prime    = x0(2);
dirX0_tpl_prime = x0(3);
Xf_tpl_prime    = x0(4);
dirXf_tpl_prime = x0(5);
lam0_2pr        = x0(6);
lamf_2pr        = x0(7);
```

DMlfun.m function file

This function runs the Simulink file “DM2_1” using the varied parameters given as the input to the function.

```
function f = DMlfun(x0)
tauf            = x0(1);
X0_tpl_prime    = x0(2);
dirX0_tpl_prime = x0(3);
Xf_tpl_prime    = x0(4);
dirXf_tpl_prime = x0(5);
lam0_2pr        = x0(6);
lamf_2pr        = x0(7);

opt = simset('SrcWorkspace', 'Current');
sim('DM2_1', [0 200], opt);
f=yout(length(yout));
```

Compute_Controls.m file

This file calculates the position commands for the quadrotor at the controller frequency.

```
% Controller speed
ctrl_t_step = .005;

% Run Simulation to get data
sim('DM3_1', [0 200])
[m_a,n_a] = size(a);
t_a_end = a(m_a,1);
t_a = 0:ctrl_t_step:t_a_end;

[m_b,n_b] = size(b);
t_b_end = b(m_b,1);
t_b = 0:ctrl_t_step:t_b_end;

% Setup Variables
tau_a = a(:,1);
x_a = a(:,4);
y_a = a(:,5);
z_a = a(:,6);

tau_b = b(:,1);
x_b = b(:,4);
y_b = b(:,5);
z_b = b(:,6);

%% Interpolate data
% Interpolate data between points at the same frequency the controller
% runs at.
x_a = interp1(tau_a,x_a,t_a,'linear');
y_a = interp1(tau_a,y_a,t_a,'linear');
z_a = interp1(tau_a,z_a,t_a,'linear');

x_b = interp1(tau_b,x_b,t_b,'linear');
y_b = interp1(tau_b,y_b,t_b,'linear');
z_b = interp1(tau_b,z_b,t_b,'linear');

%% Setup data for use in controller
% Setup a series of commands for the first waypoint
t_start = 20; %Start time for maneuver
t_a = t_a+t_start;
t_b = t_b+t_start;
t_beginning = 0:ctrl_t_step:t_start-ctrl_t_step;
z_comp = ones(1,length(t_beginning));

t_comp_a = [t_beginning' t_beginning';t_a' t_a'];
x_command_a = [t_beginning' x_a(1)*z_comp';t_a' x_a'];
y_command_a = [t_beginning' y_a(1)*z_comp';t_a' y_a'];
z_command_a = [t_beginning' z_a(1)*z_comp';t_a' z_a'];
```

```
t_comp_b = [t_beginning’ t_beginning’;t_b’ t_b’];
x_command_b = [t_beginning’ x_b(1)*z_comp’;t_b’ x_b’];
y_command_b = [t_beginning’ y_b(1)*z_comp’;t_b’ y_b’];
z_command_b = [t_beginning’ z_b(1)*z_comp’;t_b’ z_b’];
```

LIST OF REFERENCES

[1] R. Weiger, “Military unmanned aircraft systems in support of homeland security,” U.S. Army War College,Carlisle Barracks,Carlisle,PA,17013–5050 March 30th 2007.

[2] P. Wells and D. Deguire, “TALON: A universal unmanned ground vehicle platform, enabling the mission to be the focus,” Proc. SPIE 5804, 747 (2005)

[3] J. C. Grieco, M. Prieto, M. Armada, and P. G. de Santos. “A six-legged climbing robot for high payloads.” In *IEEE Int. Conf. Cont. App.,* Trieste, Italy, 1998.

[4] A. Stentz, J. Bares, T. Pilarski, and D. Stager, “The crusher system for autonomous navigation,” in *AUVSIs Unmanned Systems North America*, August 2007.

[5] A.J. Healey et al. “Collaborative unmanned systems for maritime and port security operations,” *Proceedings of the 7th IFAC Conference on Control Applications in Marine Systems*, Center for Underwater Systems and Technologies, Zagreb, Croatia, 2007, on CD.

[6] J. M. Bromley. “Evaluation of the littoral combat ship (LCS) and Spartan Scout as Information Operations (IO) Assets,” M.S. Thesis, Naval Postgraduate School, 2005.

[7] S. Lacroix, I.K. Jung, P. Soueres, E. Hygounenc, & J.P. Berry. “The autonomous blimp project of LAAS/CNRS—Current status and research challenges.” *2002 IEEE/RSJ International Conference on Intelligent Robots and Systems—IROS 2002*. Proceedings of the Workshop WS6 Aerial Robotics (pp. 35–42).

[8] “Aggressive Maneuvers for Autonomous Quadrotor Flight” [video], http://www.youtube.com/watch?v=MvRTALJp8DM, accessed in March, 2011

[9] D. Mellinger, N. Michael, & V. Kumar, (2012). “Trajectory generation and control for precise aggressive maneuvers with quadrotors.” *The International Journal of Robotics Research*.

[10] V. Kumar, N. Michael. “Opportunities and challenges with autonomous micro aerial vehicles.” *Int. Symp. On Robotics Research*, 2011

[11] M. Turpin, N. Michael, & V. Kumar, (2012). “Trajectory design and control for aggressive formation flight with quadrotors.” *Autonomous Robots*.

[12] D. Mellinger, M. Shomin, N. Michael, and V. Kumar. “Cooperative grasping and transport using multiple quadrotors.” In *International Symposium on Distibuted Autonomous Systems*, Lussanne, Switzerland, November 2010.

[13] G. Hoffman, H. Huang, S. Waslander, & C. Tomlin, “Real-time indoor autonomous vehicle test environment,” *Control Systems Magazine,* IEEE 28(2), 51–64 (2008).

[14] “Innovative UAV aircraft & aerial video systems,” http://www.draganfly.com, accessed in March, 2011

[15] “Aerospace controls laboratory at MIT,” http://vertol.mit.edu/prjinfo.html, accessed in March, 2011

[16] “AR.drone parrot,” http://ardrone.parrot.com, accessed in March, 2011

[17] Quanser Innovate Educate, “Quanser Qball-x4 user manual,” Document Number 888.

[18] “Hobby Hobby”, http://hobbyhobby.com/store/product/68199/%22Park-400-Brushless-Outrunner-Motor%2C-740Kv%22/, accessed in March, 2011

[19] J. Postel. “Transmission control protocol.” Request for Comments 793, September 1981.

[20] Natural Point, “Optitrack,” www.optitrack.com, accessed in March, 2011

[21] S.W. Paris, C.R. Hargraves: “OTIS 3.0 manual.” Boeing Space and Defense Group, Seattle (1996)

[22] J.T. Betts, W.P. Huffman: “Sparse optimal control software SOCS.” Mathematics and Engineer- ingAnalysis, TechnicalDocumentMEA-LR-085. Boeing Information and Support Services, The Boeing Company, Seattle (1997)

[23] O. Von Stryk: User’s Guide for DIRCOL (Version 2.1): A “Direct collocation method for the numerical solution of optimal control problems.” Fachgebiet Simulation und Systemopti- mierung (SIM), Technische Universität Darmstadt (2000)

[24] I.M. Ross: User’s Manual for DIDO: “A MATLAB Application Package for Solving Optimal Control Problems.” Tomlab Optimization Inc., TR 04–01.0 (2004)

[25] I. Kaminer, O. Yakimenko, V. Dobrokhodov, A. Pascoal, N. Hovakimyan, C. Cao, A. Young, et al. (2007). “Coordinated path following for time-critical missions of multiple UAVs via L 1 adaptive output feedback. Electrical engineering,” (August), 1–34.

[26] G. Milionis, “A framework for collborative quadrotor-ground robot missions,” M.S. thesis, Mech. Engr., NPS., Monterey, CA, 2011.

[27] I.D. Cowling, O.A. Yakimenko, J.F. Whidborne, & A.K. Cooke (2010). “Direct method based control system for an autonomous quadrotor. Journal of Intelligent & Robotic Systems,” 60(2), 285–316.

[28] C. Gokcek, P.T. Kabamba, and S.M. Meerkov. “An LQR/LQG theory for systems with saturating actors. IEEE Trans. Automat. Control,” 46(10):1529–1542, 2001.

THIS PAGE INTENTIONALLY LEFT BLANK

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center
   Ft. Belvoir, Virginia

2. Dudley Knox Library
   Naval Postgraduate School
   Monterey, California